



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
-----------------	-------------	----------------------	---------------------	------------------

10/066,278

01/31/2002

Paul Sheng-Chi Su

SC11654TS

5194

23125

7590

06/07/2004

EXAMINER

LU, KUEN S

FREESCALE SEMICONDUCTOR, INC.  
LAW DEPARTMENT  
7700 WEST PARMER LANE MD:TX32/PL02  
AUSTIN, TX 78729

ART UNIT

PAPER NUMBER

2177

DATE MAILED: 06/07/2004

3

Please find below and/or attached an Office communication concerning this application or proceeding.

**Office Action Summary**

Application No.

10/066,278

Applicant(s)

SU ET AL.

Examiner

Kuen S Lu

Art Unit

2177

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

- 1) ☒ Responsive to communication(s) filed on 31 January 2002.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

- 4) ☒ Claim(s) 1-20 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 1-20 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

**Application Papers**

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on \_\_\_\_\_ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)  
Paper No(s)/Mail Date 2/1-31-02.
- 4) ☐ Interview Summary (PTO-413)  
Paper No(s)/Mail Date. \_\_\_\_\_.
- 5) ☐ Notice of Informal Patent Application (PTO-152)
- 6) ☐ Other: \_\_\_\_\_.

## **DETAILED ACTION**

### **Specification**

1. Applicant is reminded of the proper language and format for an abstract of the disclosure.

The abstract should be in narrative form and generally limited to a single paragraph on a separate sheet within the range of 50 to 150 words. It is important that the abstract not exceed 150 words in length since the space provided for the abstract on the computer tape used by the printer is limited. The form and legal phraseology often used in patent claims, such as "means" and "said," should be avoided. The abstract should describe the disclosure sufficiently to assist readers in deciding whether there is a need for consulting the full patent text for details.

The language should be clear and concise and should not repeat information given in the title. It should avoid using phrases which can be implied, such as, "The disclosure concerns," "The disclosure defined by this invention," "The disclosure describes," etc.

The Abstract of the disclosure is objected to because it contains numerical references, for example, (20). Appropriate correction is required. See MPEP § 608.01(b).

### **Claim Objections**

2. Claim 10 is objected to because of the following informalities: the phrase "such as" renders the claim indefinite because it is unclear whether the limitations following the phrase are part of the claimed invention. See MPEP § 2173.05(d). Appropriate correction is required.

### **Claim Rejections - 35 USC § 103**

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a

person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

3. Claims 1-3 are rejected under 35 U.S.C. 103(a) as being unpatentable over Leong et al. (U.S. Publication 2003/0078902, hereafter "Leong") in view of Garg (U.S. Patent 6,161,152, hereafter "Garg"), and further in view of Swenson et al. (U.S. Patent 6,064,380, hereafter "Swenson").

As per Claim 1, Leong teaches the following:

"In a device that uses a JAVA programming language and a second programming language, a method of emulating more simultaneously open files in the device than is permitted by an operating system of the device" at Fig. 1 and Page 3, [0027] where APIs are implemented as a JAVA program, in a first programming language, and the server includes a JVM to convert JAVA codes to native language instructions with one language in C (Page 4, [0046]). Leong further explained the implementation of the first and second programming languages at Page 1, [0008] and Page 2, [0025] specifying JAVA and non-object programming languages.

"while running a predetermined program in the JAVA programming language, attempting to access a file stored within the device" at Page 3, [0026]-[0027] where API is a JAVA language implementation, including a JVM and at Page 3, [0033] where the interface creates a JAVA source file or accesses an XML class schema file at Fig. 5, elements 150-152.

Leong does not specifically teach on file's further operation, including **"determining whether the file is already open from prior execution"**, although Leong teaches

accessing file across the reference, for example, in steps 200 and 206 where JAVA source file is converted into executables without explicitly describing file opening status.

However, Garg **explicitly teaches opening file** at Fig. 3, steps 310-350, and col. 5, lines 27-28 by requesting opening a file and deciding next step of action depending on the status of the request.

It would have been obvious to one having ordinary skill in the art at the time of the applicant's invention was made to combine Garg's reference with Leong's because both references devoted to frequent file data manipulation (Leong: Figs. 6 and 7 on file conversion, Garg: Fig. 3 on file i/o operations), although Leong teaches application on both database and files in a multiple languages environment while Garg specifically teaches file i/o operation. The combined reference would have complimented the teachings of efficient i/o, file operation and data base data manipulation into a system for allowing application programs in different programming languages efficiently utilizes the same object oriented database management system.

Garg further teaches the rest of the claim 1 as shown in the following:

"if the file is open, access to the file proceeds" at Fig. 3, element 350 and col. 5, line 30 by performing file read/write operation;

"if the file is not open, a determination is made by the device of a number of files the device permits to be simultaneously open" at Fig. 3, element 320 and col. 5, lines 41-45 where control process determines whether the number of files opened has reached the maximum allowed;

"if the number is not exceeded by opening the file, the file is opened" at col. 5, lines 47-48 where the control process opening the file;

"if the number is exceeded by opening the file, the device emulates having more files open than the number allowed by: (1) saving a file position for at least one open file, the file position designating where a next byte in the at least one open file would be accessed" at col. 4, lines 21-25 by closing the file to free a file descriptor;

"(2) closing the at least one open file" at col. 4, lines 21-25 by closing the file to free a file descriptor; and "(3) opening the file" at Fig. 3, element 340 and col. 5, lines 47-48 by opening the file.

The combined Garg-Leong reference teaches file manipulation by a predetermined program in the JAVA programming language environment as described above.

The combined Garg-Leong reference does not specifically teach "'wherein subsequent accesses to the at least one open file that has been closed are made transparently to the predetermined program" by "closing at least one currently open file, retrieving the file position previously saved, and opening and restoring the file position for that file".

However, Swenson (col. 4, line 65.— col. 5, line 13) teaches a computer system in which completion point file positions of multimedia file presentations may be saved in persistent memory devices when a user desires to terminate a multimedia file being presented on a display device. In subsequent network and multimedia file accesses, a user is selectively able to begin play at the previously saved completion point in the

multimedia file presentation, i.e. the file position at which the user had previously terminated play.

It would have been obvious to one having ordinary skill in the art at the time of the applicant's invention was made to combine Swenson's reference with Garg and Leong's by retaining files opening and closing status, including the positions files were lastly closed because all three references are devoted to frequent file data manipulation (Leong: Figs. 6 and 7 on file conversion, Garg: Fig. 3 on file i/o operations, Swenson: Fig. 4 on file selection and positioning). The file opening, positioning and closing operations contribute most of the file i/o which would have greatly impacted system performance. In a multiple programming languages environment where files are opened and closed repeatedly due to the limit of number of files can be opened simultaneously, it would have been a much more efficient operation by recording file position right before it is closed, and making the information available for next file opening and positioning, instead of searching file positions repeatedly.

4. As per claim 2, Swenson further teaches "saving the file position in storage allocated for usage" at col. 5, lines 44-46 where the file position is saved in persistent memory or storage.

Swenson does not specifically teach the saving information is for JAVA code.

However, Leong teaches JAVA programming language program opening file at Fig. 5, elements 150 and 152, and Page 3, [0033] where JAVA program access an XML class schema file.

It would have been obvious to one having ordinary skill in the art at the time of the

applicant's invention was made to combine Swenson's reference with Garg and Leong's by providing file position information to the Leong's API programs such that the JAVA coded programs could have accessed the file's content efficiently where the interface program would have been able to improve the system performance because of speedy file record accessing.

As per claim 3, Leong teaches "implementing the second programming language as C programming language" at Page 4, [0039] by suggesting JAVA program can store data objects from different application programming languages, such as C.

5. Claims 4-8 are rejected under 35 U.S.C. 103(a) as being unpatentable over Leong et al. (U.S. Publication 2003/0078902, hereafter "Leong") in view of Garg (U.S. Patent 6,161,152, hereafter "Garg") and Swenson et al. (U.S. Patent 6,064,380, hereafter "Swenson"), as applied to claims 1-3, and further in view of Miller et al. (U.S. Patent 5,7742,902, hereafter "Miller").

As per claim 4, the combined reference of Leong, Garg and Swenson teaches emulating simultaneously opening files in a JAVA programming and a second programming languages environment as previously described for rejecting claim 1.

The combined reference does not specifically teach "storing arbitrarily long file names in a first format".

However, Miller teaches "providing a table" for "storing arbitrarily long file names in a first format" at col. 5, lines 10-15 by providing a B-tree structure for storing file name of one format and converting the name of one format to a corresponding file name of another format.



Miller further teaches the following:

“storing corresponding shortened names in a second format not exceeding a maximum number of characters” at col. 5, lines 29-31 and col. 6, lines 41-43 where short name is located at the same leaf node as the long name and the short name is of the format 8.3, a maximum of 12 characters; and

“translating from the first format to the second format” at Fig. 4 and col. 6, lines 40-67 for the translation.

It would have been obvious to one having ordinary skill in the art at the time of the applicant's invention was made to combine Miller's reference with Swenson's and the combined Garg-Leong reference by converting the arbitrarily long file names used in JAVA program into shortened names compatible with the requirement of the operating system or the application of non-JAVA language program module, and maintaining a conversion table because the system is a multiple programming language environment where a prompt lookup of file names between different programming languages would have improved the system performance.

Leong teaches “when a file is initially opened under control of the JAVA programming language, the translating being implemented transparent to any JAVA programming language application so that the JAVA programming language does not utilize the corresponding shortened names” at Fig. 5, elements 150-152 and 156 by showing the JAVA program to access an XML class file.

As per claim 5, Miller further teaches “performing the translating only when the operating system does not permit file names having a length exceeding a

predetermined maximum number of characters" at col. 3, Table A where file name length of 8.3 or longer is translated.

As per claim 6, Miller further teaches "performing the translating only when the operating system does not support first format file names" at col. 9, lines 41-44 if either long formatted or short formatted file name can be used to accurately and directly reference a file without requiring additional file name translation and at col. 3, lines 24-30 if application of previous operating system must use the restricted short format.

As per claim 7, Miller further teaches "placing a unique identifier within predetermined positions of the shortened names to identify which file is being retrieved by the operating system" at Fig. 3, elements 300-302 and col. 6, lines 40-44 and col. 8, lines 7-29 where file name is translated to 8.3 format of pre-determined 8 positions for name and 3 positions for type, and further applies conflict resolution to make the translated file name unique.

As per claim 8, Miller further teaches "using the unique identifier to bypass the translating of the first format to the second format when a predetermined file is reopened after being previously closed" at Fig. 10, elements 1000-1004 and col. 5, lines 19-28 where a known file name in the B-tree structure, there is no translation of file name needed.

6. Claims 9-13 are rejected under 35 U.S.C. 103(a) as being unpatentable over Leong et al. (U.S. Publication 2003/0078902, hereafter "Leong") in view of Garg (U.S. Patent 6,161,152, hereafter "Garg") and Swenson et al. (U.S. Patent 6,064,380, hereafter

“Swenson”), as applied to claims 1-3, and further in view of Wallman et al. (U.S. Publication 2004/0015855, hereafter “Wallman”).

As per claim 9, the combined reference of Leong, Garg and Swenson teaches emulating simultaneously opening files in a JAVA programming and a second programming languages environment as previously described for rejecting claim 1.

The combined reference does not specifically teach storing directory information into table.

However, Wallman teaches “providing a table” for “storing a directory in the table, the directory indicating a JAVA language application suite corresponding to a predetermined file” at Page 1, [0010] by providing a directory for JAVA class file where the directory is implemented as an attribute in the attribute table of the JAVA class file.

It would have been obvious to one having ordinary skill in the art at the time of the applicant’s invention was made to combine Wallman’s reference with Swenson’s and the combined Garg-Leong reference by providing a directory for JAVA class files and further implementing the directory as an attribute in the JAVA attribute table because all the references are devoted to efficiently access files or data objects (Leong: Page 1, [0006] for application to access data objects of other applications, Garg: col. 5, lines 41-45 for accessing and opening a plurality of files, Swenson: col. 2, lines 12-15 for improved file presentation and effective file access and Wallman: Page 1, [0008] for efficiently accessing components of object files). The combined reference would have allowed users of Leong’s system to open and access a plurality of JAVA application files in an efficient fashion because of the implementation of file directory into an attribute

table which would have been able to avoid a search on a various number of components of variable lengths from a plurality of object oriented language files.

Wallman further teaches "using the directory in the table to distinguish files of the same name belonging to different JAVA language application suites" at Fig. 4, elements 402-404 by using the marked components to differentiate the class file not to be loaded into JVM.

As per claim 10, Swenson teaches "using storage allocated for use by JAVA code to contain variables such as file position needed by the second programming language" at col. 4, line 65 – col. 5, line 13 by showing a computer system in which completion point file positions of multimedia file presentations may be saved in persistent memory devices when a user desires to terminate a multimedia file being presented on a display device.

7. As per claim 11, Swenson teaches "using the variables to track the current position within an open file and to reopen and reposition the file if the file is temporarily closed" at col. 4, line 65 – col. 5, line 13 by showing in a subsequent network and multimedia file accesses, a user is selectively able to begin play at the previously saved completion point in the multimedia file presentation, i.e. the file position at which the user had previously terminated play.

As per claim 12, Garg teaches "using automatic memory management features of the JAVA programming language including garbage collection to reclaim storage when the file is no longer in use" at col. 1, lines 35-40 by describing a fully functional UNIX

operating system where memory management, including garbage collection, is an essential component of a 'fully functional UNIX operating system.

As per claim 13, Leong teaches "implementing the device as a portable, wireless device" at Page 2, [0024] where hand-held computer is able to connect to the internet via wireless connection.

8. Claim 14 is rejected under 35 U.S.C. 103(a) as being unpatentable over Leong et al. (U.S. Publication 2003/0078902, hereafter "Leong") in view of Garg (U.S. Patent 6,161,152, hereafter "Garg"), and further in view of Swenson et al. (U.S. Patent 6,064,380, hereafter "Swenson").

As per Claim 14, Leong teaches the following:

"interfacing a file system in a device that uses both a JAVA programming language and another programming language, comprising: selecting a predetermined operating system" at Fig. 1 and Page 3, [0027] where APIs are implemented as a JAVA program, in a first programming language, and the server includes a JVM to convert JAVA codes to native language instructions with one language in C (Page 4, [0046]). Leong further explained the implementation of the first and second programming languages at Page 1, [0008];

"overlying a JAVA virtual machine for executing programs within the operating system in the another programming language" at Page 2, [0027] where JAVA Virtual Machine is implemented on the server system to convert JAVA byte-codes to instructions in the native machine language of the database server which supports non-JAVA application programming languages.

“overlying libraries to the JAVA virtual machine and the file system interface layer, the libraries running in the JAVA programming language” at at Page 2, [0027] where JAVA Virtual Machine is implemented on the server system to convert JAVA byte-codes to instructions in the native machine language and the database interface accesses non-JAVA application interfaces to manipulate data within non-JAVA objects in a multiple non-JAVA application programming languages.

Leong does not specifically teach on file's operation, including “using the file system interface to determine if a maximum number of open JAVA files has been exceeded, and if so, storing position information of at least one open JAVA file in a currently open JAVA application prior to closing the at least one open JAVA file”, although Leong teaches accessing file across the reference.

However, Garg teaches opening file at Fig. 3, steps 310-350, and col. 5, lines 27-28 by requesting opening a file and deciding next step of action depending on the status of the request, using control process to determines whether the number of files opened has reached the maximum allowed at Fig. 3, element 320 and col. 5, lines 41-45, and closing the file to free a file descriptor at col. 4, lines 21-25.

It would have been obvious to one having ordinary skill in the art at the time of the applicant's invention was made to combine Garg's reference with Leong's because both references are devoted to database application and i/o operation. The combined reference would have combined the teachings of efficient i/o, file operation and data base data manipulation into a system for allowing application programs in different

programming languages efficiently utilizes the same object oriented database management system.

The combined Garg-Leong reference does not specifically teach “storing position information of at least one open JAVA file in a currently open JAVA application prior to closing the at least one open JAVA file” and “subsequently reopening the at least one open JAVA file by restoring the position information transparent to any JAVA application.

However, Swenson (col. 4, line 65 – col. 5, line 13) teaches a computer system in which completion point file positions of multimedia file presentations may be saved in persistent memory devices when a user desires to terminate a multimedia file being presented on a display device. In subsequent network and multimedia file accesses, a user is selectively able to begin play at the previously saved completion point in the multimedia file presentation, i.e. the file position at which the user had previously terminated play.

It would have been obvious to one having ordinary skill in the art at the time of the applicant's invention was made to combine Swenson's reference with Garg and Leong's by retaining files opening and closing status, including the positions files were lastly closed because all three references are devoted to frequent file data manipulation. The file opening, positioning and closing operations contribute most of the file i/o which would have greatly impacted system performance. In a multiple programming languages environment where files are opened and closed repeatedly due to the limit of number of files can be opened simultaneously, it would have been a much more

efficient operation by recording file position right before it is closed, and making the information available for next file opening and positioning, instead of searching file positions repeatedly.

9. Claim 15-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Leong et al. (U.S. Publication 2003/0078902, hereafter "Leong") in view of Garg (U.S. Patent 6,161,152, hereafter "Garg") and Swenson et al. (U.S. Patent 6,064,380, hereafter "Swenson"), as applied to claim 14, and further in view of Wallman et al. (U.S. Publication 2004/0015855, hereafter "Wallman").

As per claim 15, the combined reference of Leong, Garg and Swenson teaches emulating simultaneously opening files in a JAVA programming and a second programming languages environment as previously described for rejecting claim 1.

The combined reference does not specifically teach storing directory information into table.

However, Wallman teaches "providing a table" for "storing a directory in the table, the directory indicating a JAVA language application suite corresponding to a predetermined file" at Page 1, [0010] by providing a directory for JAVA class file where the directory is implemented as an attribute in the attribute table of the JAVA class file.

It would have been obvious to one having ordinary skill in the art at the time of the applicant's invention was made to combine Wallman's reference with Swenson's and the combined Garg-Leong reference by providing a directory for JAVA class files and further implementing the directory as an attribute in the JAVA attribute table because all the references are devoted to efficiently access files or data objects (Leong: Page 1,



[0006] for application to access data objects of other applications, Garg: col. 5, lines 41-45 for accessing and opening a plurality of files, Swenson: col. 2, lines 12-15 for improved file presentation and effective file access and Wallman: Page 1, [0008] for efficiently accessing components of object files). The combined reference would have allowed users of Leong's system to open and access a plurality of JAVA application files in an efficient fashion because of the implementation of file directory into an attribute table which would have been able to avoid a search on a various number of components of variable lengths from a plurality of object oriented language files.

Wallman further teaches "using the directory in the table to distinguish files of the same name belonging to different JAVA language application suites" at Fig. 4, elements 402-404 by using the marked components to differentiate the class file not to be loaded into JVM.

As per claim 16, Swenson teaches "using storage allocated for use by JAVA code to contain variables such as file position needed by the second programming language" at col. 4, line 65 – col. 5, line 13 by showing a computer system in which completion point file positions of multimedia file presentations may be saved in persistent memory devices when a user desires to terminate a multimedia file being presented on a display device.

As per claim 17, Swenson teaches "using the variables to track the current position within an open file and to reopen and reposition the file if the file is temporarily closed" at col. 4, line 65 – col. 5, line 13 by showing in a subsequent network and multimedia file accesses, a user is selectively able to begin play at the previously saved completion

point in the multimedia file presentation, i.e. the file position at which the user had previously terminated play.

10. Claims 18-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Leong et al. (U.S. Publication 2003/0078902, hereafter "Leong") in view of Garg (U.S. Patent 6,161,152, hereafter "Garg") and Swenson et al. (U.S. Patent 6,064,380, hereafter "Swenson"), as applied to claim 14, and further in view of Miller et al. (U.S. Patent 5,7742,902, hereafter "Miller").

As per claim 18, the combined reference of Leong, Garg and Swenson teaches emulating simultaneously opening files in a JAVA programming and a second programming languages environment as previously described for rejecting claim 1.

The combined reference does not specifically teach "storing arbitrarily long file names in a first format".

However, Miller teaches "providing a table" for "storing arbitrarily long file names in a first format" at col. 5, lines 10-15 by providing a B-tree structure for storing file name of one format and converting the name of one format to a corresponding file name of another format.

Miller further teaches the following:  
"storing corresponding shortened names in a second format not exceeding a maximum number of characters" at col. 5, lines 29-31 and col. 6, lines 41-43 where short name is located at the same leaf node as the long name and the short name is of the format 8.3, a maximum of 12 characters; and

"translating from the first format to the second format" at Fig. 4 and col. 6, lines 40-67 for the translation.

It would have been obvious to one having ordinary skill in the art at the time of the applicant's invention was made to combine Miller's reference with Swenson's and the combined Garg-Leong reference by converting the arbitrarily long file names used in JAVA program into shortened names compatible with the requirement of the operating system or the application of non-JAVA language program module, and maintaining a conversion table because the system is a multiple programming language environment where a prompt lookup of file names between different programming languages would have improved the system performance.

Leong teaches "when a file is initially opened under control of the JAVA programming language, the translating being implemented transparent to any JAVA programming language application so that the JAVA programming language does not utilize the corresponding shortened names" at Fig. 5, elements 150-152 and 156 by showing the JAVA program to access an XML class file.

As per claim 19, Miller further teaches "placing a unique identifier within predetermined positions of the shortened names to identify which file is being retrieved by the operating system" at Fig. 3, elements 300-302 and col. 6, lines 40-44 and col. 8, lines 7-29 where file name is translated to 8.3 format of pre-determined 8 positions for name and 3 positions for type, and further applies conflict resolution to make the translated file name unique.

As per claim 20, Miller further teaches "using the unique identifier to bypass the translating of the first format to the second format when a predetermined file is reopened after being previously closed" at Fig. 10, elements 1000-1004 and col. 5, lines 19-28 where a known file name in the B-tree structure, there is no translation of file name needed.

### ***Conclusions***

11. The prior art made of record

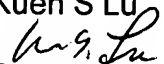
- A. U.S. Publication 2003/0078902
- B. U.S. Patent 6,161,152
- C. U.S. Patent 6,064,380
- D. U.S. Patent 5,745,902
- E. U.S. Publication 2004/0015855

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.


- F. U.S. Patent 6,519,594
- G. U.S. Patent 5,765,169
- H. U.S. Publication 2003/0088783
- I. U.S. Patent 6,611,858
- J. U.S. Patent 6,301,583
- K. U.S. Publication 2002/0087571

Art Unit: 2177

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kuen S Lu whose telephone number is 703-305-4894. The examiner can normally be reached on 8 AM to 5 PM, Monday through Friday. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Breene can be reached on 703-305-9790. The fax phone number for the organization where this application or proceeding is assigned is (703) 872-9306. Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is 703-305-3900.

Kuen S Lu  
  
Patent Examiner

May 28, 2004

  
SRIRAMA CHANNAVEJULA  
PRIMARY EXAMINER